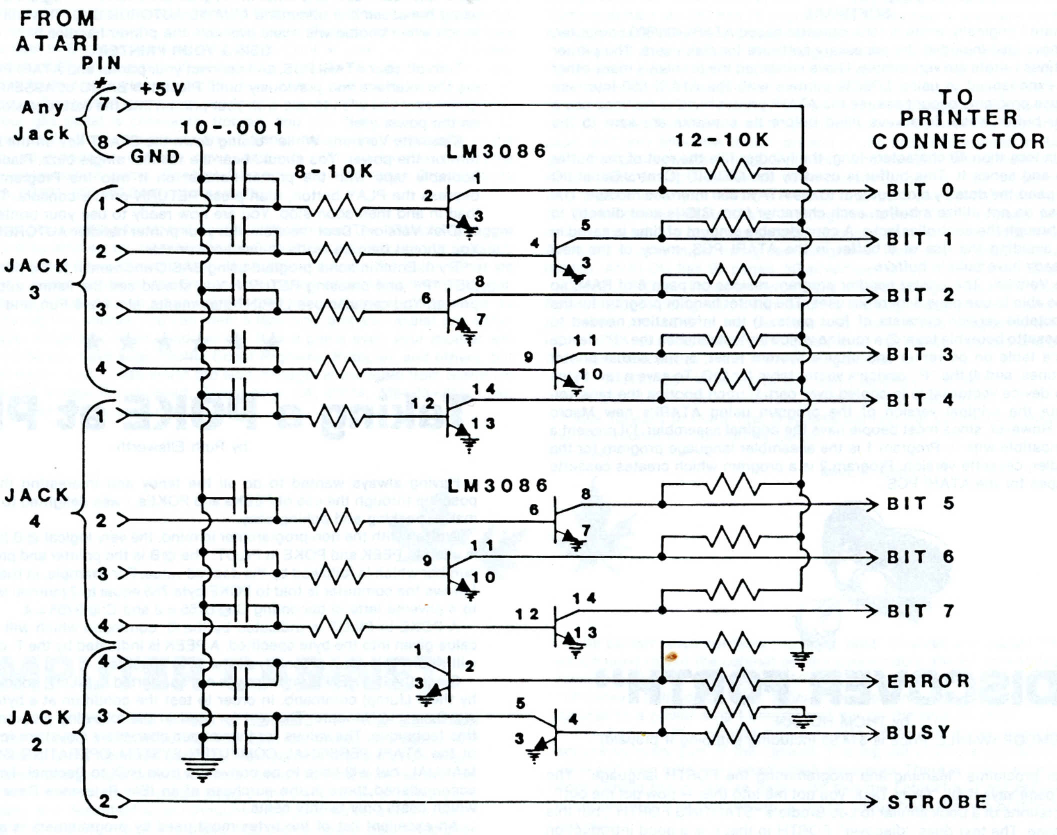
ATARI COMPUTER ENTHUSIASTS

3662 Vine Maple Dr. Eugene OR 97405

November, 1982

Mike Dunn & Jim Bumpas, Editors



Printer/Joystick Interface
by Ed Chastain

Connecting an Epson MX-80 Printer to an ATARI

by Ed Chastain

This article discusses connecting an EPSON MX-80 (or any other) printer to the front controller jacks of an ATARI Personal Computer System (PCS). It explains the necessary hardware and software required to accomplished this. Recently, a friend of mine looked into purchasing a printer for his ATARI PCS but, he didn't want to purchase an ATARI 850 interface module. I told him it is possible to connect the printer to the front controller jacks of his ATARI computer. After he decided to purchase the EPSON printer, it was my task to build an interface and write the necessary printer handler for him.

There are currently several devices which connect to the controller jacks, including: another printer interface, various EPROM programmers, and a direct connect modem.

The Macrotronic printer interface uses two of the controller jacks, but with only seven data bits, the eighth bit is used as the strobe, and the trigger inputs are used for busy and error status. This is fine with most printers, but many of the new ones can use all eight bits. I chose to use three controller jacks because I wanted all eight data bits, since the MX-80 with graphics can utilize the eighth bit.

HARDWARE

The hardware I used consists of the following parts: two LM3086 transistor arrays; twenty 10K Ohm, quarter Watt resistors; ten 0.001 microfarad capacitors; two 15 wire cables (lengths depend upon your needs); three 9-pin connectors (compatible with ATARI controller jacks); printer connector (compatible with your printer); a printed circuit board kit, or breadboard, or experimentor plugin breadboard, or anything you choose to build your circuit on; a small plastic box of a size compatible with your circuit board (optional). The schmatic diagram for the circuit is shown in the accompanying Figure.

SOFTWARE

The software I originally wrote is for a cassette based ATARI 400/800 computer. However, I have also included the necessary software for disk users. The printer handler routines I wrote are very simple. I have corrected the problems many other users have experienced in using EPSON printers with the ATARI 850 interface module. These problems occur because the ATARI printer handler routines use a 40 character buffer which is always filled before its contents are sent to the printer.
If your line is less than 40 characters long, the handler fills the rest of the buffer with blanks and sends it. This buffer is used by the CIO/SIO (Central/Serial I/O routines) to send the data by block output to the ATARI 850 interface module. The routines I use do not utilize a buffer, each character from CIO is sent directly to the printer through the controller jacks. A considerable amount of time is saved in printing by avoiding the use of a buffer in the ATARI PCS, many of the new printers already have built-in buffers.

(Cassette Version). The printer handler program resides on page 6 of RAM, so you won't be able to use page 6 for other uses. The printer handler program for the cassette bootable version consists of four parts: 1) the information needed to make the cassette bootable tape; 2) a routine which (re-)establishes the "P" device in the device table on power-up and after a system reset; 3) the actual printer handler routines; and 4) the "P" device's vector table for CIO. To save a few bytes of RAM the device vector table is copied over part 1) upon booting the tape version. I wrote the original version of the program using ATARI's new Macro Assembler. However, since most people have the original assembler, I'll present a version compatible with it. Program 1 is the assembler language program for the printer handler, cassette version. Program 2 is a program which creates cassette bootable tapes for the ATARI PCS.

(Disk Version). The disk version of the printer handler also resides on page 6 of RAM. Program 3 is the disk version of the printer handler. It consists of four parts: 1) the handler vector table; 2) a routine which (re-) establishes the "P" device in the device table on power-up and after a system reset; 3) the printer handler routines; and, 4) a routine which saves DOS's initialization vector and replaces it with the printer's initialization vector.

PROCEDURE FOR MAKING TAPE OR DISK VERSION

(Cassette Version). The procedure to make a cassette bootable tape with the printer handler on it is relatively simple. First, place the assembler cartridge in the ATARI PCS and enter Program 1. You may wish to save this program to a tape to avoid re-entering it at some later time. Next, assemble the program using the ASM asssembler command. Enter the NEW command, and then type in Program 2. Again, you may wish to save Program 2 for some later time. Assemble Program 2 using the ASM command. Next, enter into the debugger by typing in BUG. Using the G command, G4000 run Program 2. You should hear the familiar two buzzes, set up a blank tape for recording by moving the tape ahead past the leader. Place the blank tape into the Program Recorder, depress the RECORD and PLAY buttons, then press RETURN on the console. After the Program Recorder stops, rewind the tape. You now have a cassette bootable tape with a new printer handler which utilizes the front controller jacks instead of the 850 interface module.

(Disk Version). The procedure to make an AUTORUN.SYS file with the printer hanlder on it follows. First, place the assembler cartridge in the ATARI PCS, then boot DOS 2.0. Format a blank disk and write the DOS files to it. Return to the assembler cartridge, and enter in Program 3. Assemble Program 3 to a disk file by using the assembler command ASM,,#D:AUTORUN.SYS. This will give you a disk which when booted will make available the printer handler.

USING YOUR PRINTER

Turn off your ATARI PCS, and connect your printer and ATARI PCS together using the interface you previously built. Place the BASIC or ASSEMBLER cartridge (whichever you wish to use with your printer) into the cartridge slot, but don't turn on the power yet!

(Cassette Version). While holding down the START key on the ATARI console, turn on the power. You should hear the familiar single buzz. Place your cassette bootable tape with the printer handler on it into the Program Recorder and depress the PLAY button, then press RETURN on the console. The tape should load in and then soon stop. You are now ready to use your printer.

(Disk Version). Boot the disk with your printer handler AUTORUN.SYS file on it. You should then be ready to use your printer.

Try it. Enter in some program using BASIC and send it to the printer by typing in LIST "P:" and pressing RETURN. You should see the listing appearing on your printer. You can also use LPRINT statements, etc. Have Fun, and Happy Printing.

★ ★ ★ ★ ★ ★

"DISCOVER FORTH"

by THOM HOGAN

(OSBORNE/MCGRAW-HILL. Price is $14.95 including shipping if prepaid)

The cover proclaims "learning and programming the FORTH language". The dedication page says it all, "Okay, Dick, you got me into this — now get me out!". I had expectations of a book similar to Leo Brodie's "STARTING FORTH", but this is not the case. The text does "discover" FORTH in that it is a good introduction to the FORTH language. The book may be considered a complement to a complete text on programming in the FORTH language. It is not a usable single source of information on programming in FORTH. If you must have a complete library of all the books written on FORTH language, pick it up and put it on the shelf.

—LARRY SANO

Taking a POKE at PILOT

by Ruth Ellsworth

Having always wanted to do all the fancy and interesting things which are possible through the use of PEEKs and POKEs, I was delighted to discover PILOT makes peeking and poking easy.

Written with the non-programmer in mind, the very logical @B (at byte) notation is used to PEEK and POKE in PILOT. The @B is the pointer and preceeds the byte number which is followed by the desired value. For example, in the program which follows the computer is told to POKE byte 755 equal to 2 (normal letters), and later to 4 (inverse letters) becoming C:@B755 = 2 and C:@B755 = 4.

A POKE in PILOT is indicated by the C: command which will "Compute" the calue given into the byte specified. A PEEK is indicated by the T: command which will return the value presently at the byte specified.

Since the IF THEN terminology is not supported in PILOT, conditions are tested by the J (Jump) command. In order to test the condition at a byte the command J(@Bxxx = x):*Module. The test for start in the following program demonstrates this technique. The values for the various characters (keys) are found on page 50 of the ATARI PERSONAL COMPUTER SYSTEM OPERATING SYSTEM USER'S MANUAL, but will have to be converted from HEX to Decimal. This can be easily accomplished through the purchase of an IBM Reference Data Card from LCC which costs only twenty cents.

An excellent list of the bytes most used by programmers is available in THE MASTER MEMORY MAP by Educational Software. I have not had any trouble using the ones I have tried in the standard graphics mode, but did find that some, including the small letter value did not work in the oversize test mode. The Technical Notes for PILOT are available from ATARI, and all the values and bytes indicated work easily. I recommend all programmers using PILOT obtain a set of those notes.

Take a POKE or PEEK at PILOT, it is fun and surprising what can be done!

*** PROGRAMMING HINT FOR PILOT BEGINNERS**

To use a module you have used before and don't want to retype when it is buried in another program:

Load the program
List just the numbers of the module
Type NEW
Either change the numbers and type return, or type return through each line and then type REN.

Benioff At Large

(The Big Guy Goes To USC)
by Marc Russell Benioff

Hello, and greetings from The University Of Southern California (USC). Over the past four months, this column has had three names, and three different styles. I have now decided to keep it "Benioff at Large", and write it the way I feel is correct. The reviews will begin to get longer, but they will be less in number. Only the programs and information which I feel is absolutly pertinent will appear. If it is pertinent, it will appear. The column this month reviews four games and two magazines. "Two Magazines?", you ask. Yes, Antic #4 and ANALOG #8. This month, the column takes another turn. Cross into "The Benioff Zone".

"Marc, you are not going to believe this, my computer is talking!", shouted the professional Atari programmer. "Sam, Sam, Sam, he is talking to me!", he screamed. "O.K., it is O.K.", I said back. Little did I know what he was talking about. **SAM** is The Software Automatic Mouth, a new product from **Don't Ask Software**. If you remember correctly, Don't Ask made a big hit first with **ABUSE**, and now they have released SAM. SAM is a voice synthesiser. It uses the television to talk. To answer many questions at once, it only requires an 800, a Disk Drive, and a TV. No other hardware is required. For $60, you can own an excellent voice synthesiser. SAM is a machine language subroutine which is accessed through BASIC. This means you can make your own programs talk. Included with SAM is several demos, speeches (i.e. Gettysburg Add., and The Pledge), and excellent instructions allowing anyone to use it. Is there anything bad about SAM? Yes, there is one small item, the screen blanks out while he talks. But this is the only weak point. Soon, there will be a singing version which interacts with the music composer, according to Don't Ask. How can I not give this my highest rating, it is a technical masterpiece! Contact Don't Ask At: 2265 Westwood Boulevard, Los Angeles, CA, 90064. 213-397-8811. You won't regret it!

It is not just another game. First there was **Match Racer**, now there is **Baja Buggies!** If you have ever seen **Turbo** by Sega in the arcades, you understand the complexity of a 3-D driving game. It isn't quite a "Turbo", but it comes pretty damn close! Baja Buggies is a 3-D driving game that takes place in the desert. it is in full 3-D, and even has scrolling mountains to add to the effect. It is a one player game, and allows three course options, and two skill levels (Pro and Am). When you take the driver's seat in Buggies, you see your car in front of you. You begin number Eighty in a line of Eighty Buggies. Your job is to climb to number one by the end of the course. Dodging buggies, controlling your speed, and using your brakes, all come to use in Buggies. A game well worth the money. You drive by using your joystick. Your gas pedal is constantly floored, and you can use your brakes to slow yourself down. If you are a poor driver, knock other cars, hit the side of the course, etc., your buggie will break down, and you will have to start over. Baja Buggies is an excellent game. Contact Gamestar at: 1302 State Street, Santa Barbara, CA, 93101, 805-963-3487. It's Incredible.

We have all been waiting for it, and now it is finally here. The "Official" **Frogger** by Sega, as programmed by John Harris for Sierra On-Line, Inc. Yes, it is exact. Except, there is not a two player option. Well, this is beside the point. What we have here is a game worthy of the On-Line reputation of excellance! In Frogger, you must guide a frog through a busy street, over a beach, and through a river to his home. during this course, you can meet a lady frog, and also assist her. Other perils include: alligators, otters, and worms. It is a game even your mother will enjoy. Of course you have seen Pacific Coast Highway, Preppie!, and others, but this is the Frogger. And, it was worth the wait. Frogger is available from Sierra On Line, Inc: 36575 Mudge Ranch Road, Coarsegold, CA, 93614, 209-683-6858. I can hardly wait to see **Ultima**, their next release!

Dr. Goodcode's Cavern is a new adventure game from Gebelli Software, Inc. I briefly have talked about this game before, but I think an in-depth review is worth it. CAVERN is an all text game. But, it isn't. It involves color sound, all joystick control, and yet, it is a text adventure. How is this possible? Because it is this way: It is the reason for its outstanding playability. The creative mazes (random every time), interesting prompts, and fantastic battle sequences, throw it into a realm of outstanding adventuring. It is not an adventure like the Scott Adams or On-Line type. It sets its own realm. You explore, can save your game, and all of this takes place in BASIC! Yes, BASIC! I love it, and so should you! 48K and disk are required. Contact Gebelli at: 1787 Tribute Road, Suite G, Sacramento, CA, 95815, 916-925-1432. I like Gebelli, they are different!

Before I begin my reviews and comparisons between ANALOG AND ANTIC, I think I should give you my bias towards each. I have talked to Jim Capparell many times, and have discussed article possibilities, but have never published with them. I have been printed twice in ANALOG, number 6 and 7. In number 6, I was a contributing editor. I have NOT been paid for either printing (10/25), and it has been six months since the first printing. Payment was agreed to be paid at time of printing. I consider both publications a positive force in the ATARI spectrum. I will try to keep a non-biased view, and attempt to give the reader the good and bad points of each.

Antic was first released at the West Coast Computer Fair in March. Since then they have made precise bi-monthly printings, and have maintained a publication on slick paper with exceptional artwork on the covers (issues 3 & 4). Antic wants to be the ATARI resource, and I think they are doing a very good job. Antic is laid out in a professional magazine style. Contents aside, the magazine appears to be clean cut, professional, and extremely colorful! Antic "The sound issue" features articles from simple Atari Music and sounds, to how to use sound in cassette programs. Featured writers are Jerry White and Ed Rotberg (The finest Atari sound creators ever). Antic also diverts into reviews of four new products. ANTIC also has games from Stan Ockers, and John Magdziarz. Antic also has a section on Turtle graphics. ANTIC has 78 pages, 56 advertisements, and sells for $2.50. Antic also has many other fine articles in issue 4.

A.N.A.L.O.G. also is an ATARI magazine. Analog has no theme for issue 8. On the cover is a picture of an 800 and a monitor, with a single hand, and white lines from the keys the hand pushed. ANALOG recently began using slick paper (issue 7), and also has a quality look. There are no colored pages or colored outlines, like ANTIC. ANALOG offers a audio program similar to ANTIC's, but theirs is ½ a page, and only tells how to turn off and on the program recorder in Basic. ANTIC goes into detail, while ANALOG allows a shallowness in this field. ANALOG has an excellent in depth review on three budget programs. This review goes further into depth than any of ANTIC's reviews. ANALOG and ANTIC both have a review on The Voicebox. ANALOG's is 1½ pages, and ANTIC's is ½ page. ANALOG has articles on mixing graphics modes, floating point numbers, and new products. ANALOG excells in providing user programs. They offer 8 programs, and several demos. ANALOG has 96 pages, 50 advertisements, and sells for $2.50

CONCLUSIONS. These are two totally different magazines. ANTIC is "The Atari Resource", providing valid information to the beginner and expert on the theme of the issue. ANALOG is for the educated user. It has programs, less beginner information, and has a less slick appearance, but offers less ads and promotions. Only you can choose, both are fine magazines.

ANTIC: 297 Missouri Street, San Francisco, CA 94107
ANALOG: 565 Main Street, Cherry Valley, MA 01611

I look forward to many fine programs coming out before Christmas. I will see you next month in "Benioff At Large"
Sincerely,

Marc R. Benioff

COMPUTER CABINET

I just bought one of the finest pieces of furniture I've ever seen. Paul Armstrong, owner of ACO WOODWORKS, 755 East 22ndAvenue, Eugene, OR 97405, (503) 342-4684, is the father of one ACE member and the son of another. I learned he has been designing a computer work station for and with his mother. So I approached him with some very rough ideas of my own. I drew a little picture on a scrap of paper and described what I thought were good dimensions to use. I wanted an integrated cabinet in which to place ALL the computer gear I had been using which was scattered around a large part of a room.

Paul produced a final design and built from it a solid oak cabinet, mounted on casters for easy movement from room to room. The overall dimensions are 40" high, by 32" wide, by 24" deep. The cabinet can be described as consisting of 3 "sections". The top section holds the printer which is accessible from a lid opening the top. The front is a slanted copy holder. The copy board is hinged on the top and is lifted to access the control switches in the front of the printer. To the right of this copy board is a drawer for disk storage.

Immediately below is a section with a door which swings down and provides a surface upon which to roll out the console for easy typing. This door/platform is 28" from the floor. I have the console, disk drive and modem in this compartment. There is room for a dual disk drive.

The bottom-most section has two swinging cabinet doors opening into two shelves. I store game accessories, documentation and other books, some supplies, and other miscellaneous things down here. The top shelf also contains the interface.

The bottom shelf contains the "rat's nest" of wires and cables. The only wires which need to exit the cabinet are the power cable, the RF jack to the TV, and the telephone line to the modem. All other cables and wires are routed through a "chimney" behind the disk storage drawer and extending through the entire cabinet to a power strip I bought and put on the bottom shelf.

The back of the cabinet has two sets of double doors, completing the full access to every part of the cabinet. The top set of doors opens onto the printer area, and also onto the console/disk drive area. There is a 10" wide slit cut in one door behind the printer so paper can exit the cabinet and not accumulate inside while printing. The bottom set of doors opens onto the two bottom shelves from the rear.

The workmanship of the cabinet is excellent. Paul tells me he is willing to build more of these, and will make modifications to order. For a model like the one I have (perhaps with a few minor modifications) his price is $495. He takes a few weeks to build it, give it an oil finish, and deliver it (in Eugene or nearby). If the cabinet must be shipped, it will take longer and shipping must be paid.

In the few days I've had the cabinet, I've noticed many advantages over the way in which I used the computer equipment before. All the equipment fits into floorspace 32"x 24", and can be moved easily. The cabinet is an attractive piece of furniture which looks good anywhere in the house. The printer is closed up and almost silent! All the equipment can be closed up from sight and away from dust. For the first time, I really have a convenient method for reading copy as I type it into the console. And the disk storage drawer is the best method I've yet used to store disks.

If you're interested to inquire further, I recommend you contact Paul Armstrong soon. He's only one man, and can only make so many in a given time.

—Jim Bumpas, Co-Editor

Brian's Arcade

About one week ago Ihor from Synapse Software sent me Claim Jumper, Protector II, and Picnic Paranoia.These are all new games from Synapse and they are all very good.

Protector II, by Mike Potter is the second version of Protector released by Synapse but the third version overall (the first version was released by Crystal a long time ago). Protector II is pretty much the same as Protector I with a few variations. In Protector I there is no gravity. In Protector II if you move your Omicron Needlefighter to the top of the screen you will slowly start to fall to the ground which means there is gravity. Another very good change in this version is: When you have a person clinging onto your ship you can shoot and the people will not fall off of the needlefighter. This is sort of a problem though because when you want to drop your man on a building you have to drop him right on top of the building instead of going on top of the screen and pressing the joystick button as you do in Protector I. Something else I like is: Instead of flying to the right automatically in Protector II you can fly to the left and you will be very close to the city where you save your people once and for all. In Protector II the game enables you to fly through the buildings which is very helpful because in Protector I, I always died because I kept hitting the buildings. Protector II has much better graphics than regular Protector. All I can say is: If you liked the other Protector you will love Protector II.

Claim Jumper—by Gary Chang

Claim Jumper is an old western shoot'em up game with a couple things which make it very fun. Claim Jumper is a two player game. Each person contols an old miner. The miners try to run and get the gold and turn it into the bank and then the bank will give you money for the gold. Whoever gets the money from the bank first then trys to take it to his bank. While taking it to your bank you have to avoid getting shot by the other angry miner or running into houses, cactusses, snakes, or tumbleweeds. Claim Jumper is the best two player game, where you play at the same time, on the market. The game is so good I even got my dad to play with me and he hates playing games.

Picnic Paranoia—by Russ Segal

In Picnic Paranoia you are George. George is having a picnic. Suddenly George finds he is being invaded by ants, jumping spiders and flying wasps. You are armed with your faithful fly swatter and you have to try to kill off the ants, spiders and wasps before they get your food. The only way you can stop them is by killing them with your flyswatter. Picnic Paranoia has outstanding graphics and awesome sound. The game has a one or two player option and it has an option where you can have a picnic at night or in the daytime. Overall, Picnic Paranoia is a very good and very cute game which will give you hours of enjoyment.

All of these games are available from Synapse Software and they are all very good.

Metor Storm—by Jon Atack

Royal Software in Eugene was kind enough to give us their new game Meteor Storm. Meteor Storm is kind of like Protector. The object of the game is to rescue all your people form a dome covered city. Sounds easy, but it's not because there are a couple of problems. There are meteors going across the TV screen and you can not run into them so you have to shoot them. Also there are space mines and if they run into a meteor they will explode. If you are around them when they explode you will die.

Meteor Storm has very good graphics and outstanding sound. When you start to louse the game up you will hear the theme from Raiders of the Lost Ark. Meteor Storm is written by Jon Atack and L.J. Knoll and is available from Royal Software in Eugene.

Until Next month this is Brian Dunn saying happy arcading!!

ERACE
GOOD NEWS FOR EDUCATORS!

ATARI has made a fotune producing and marketing some of the world's greatest microprocessor based games. Some people think games is all Atari does. In the minds of some, the ATARI 400/800 computers are "fancy game machines".

I didn't know that when I bought my ATARI 800. At the time, I was a high school English teacher in a small country school who wanted to see what the microcomputer could add to my classroom. The school district had no desire to buy computers, so I decided to buy one and teach myself to operate it. After careful and extensive research, I chose ATARI for the many reasons we ATARI owners are well aware of by now.

But in the world outside ATARILAND, the stigma continues. Many would-be computer owners won't even consider Atari.

How do we erase the stigma? Well, this is one reason why several of us A.C.E. members in Eugene have named our new education group **ERACE**. We hope to show educators what ATARI has to offer.

EDUCATIONAL RESEARCHERS of A.C.E. is beginning its third month, and already great things are taking shape. At our October meeting, Bob Browning, whose interest, along with mine, is designing and creating educational software, presented some of his findings about current theory in program design. His research is on-going and will be of importance to the group.

For our November meeting, Ruth Ellsworth is doing a special presentation on ATARI Pilot, demonstrating how it can be used in the home by any interested parent to supplement a child's school work. Ruth, as you probably know, has written several programs in Pilot and is our resident expert.

Having been on my own (just me and my ATARI) for almost one year and not even knowing about A.C.E., I sympathize with you courageous people out there in the wilds who have plunged into the exciting world of microcomputing. I have great hopes ERACE will meet some of the educational needs we all have whether educators, concerned parents, or persons interested in self-education.

We plan to prepare a special article or letter to the editor which we will send to various computer magazines in order to let more ATARI owners know about ERACE. Included in the letter will be a statement of our goals, which are:

1. To locate and identify available educational software;
2. To evaluate and catalog the software;
3. To design programs and to submit ideas intended to fill gaps in the educational market;
4. To inform the public through the monthly A.C.E. Newsletter;
5. To help train parents, teachers and children in the use of educational software in the home;
6. To document software which is poorly documented.

We will also make a request for input from anyone interested in helping or being helped. If you know of educational software currently being used on the ATARI 400/800 system, including your own creations, we want to know about it. We will prepare honest reviews based on actual use by young people in the age categories suggested by the producer. In the event we find the program is deficient, we will not review it but will instead send suggestions to the producer.

The next meeting of ERACE is Tuesday, December 7, 1982. For further information, contact our chairperson, Alice Erickson (687-1133), who is currently working on a doctorate in educational psychology, Larry Gold (686-1490), our software contact who is using his ATARI to write a novel (his wife is also writing a non-fiction book on the same machine!), or me, DeLoy Graham. I am presently working on a masters' degree in computer science education. We want to serve your needs.

—Russell DeLoy Graham

Reviews and News
by Mike Dunn, Co-Editor

Yesterday I received the brand new **Mosaic** 64K RAM board for the Atari 400. (Mosaic POB 708, Oregon City, OR 97045, $250). This not only expands your memory, but allows you to use 52K for most applicatations, and gives you the ability to bank-select 4 4K banks of memory. To install the board, you first take the Atari 400 completely apart, solder two wires from a cable, install the new board, change a chip and connect a ribbon cable. The instruction booklet is reasonably easy to follow and complete. You then put it all back together, and hope it works. Plan about an hour; I had no problems, but I've taken my 400 apart many times testing various memory boards, etc. This will of course void your warranty on your Atari.

It worked the first time. With a BASIC cartridge in, you turn your Atari on, and use ?FRE(0). You still show 37,900, but then with a simple POKE, you increase it to read 49,934- some of it being used by the cartridge, of course. With other programs which do not use the cartridge slot, the extra memory is automatic. In the programs I tried, the indicated memory available is increased by about 4K in VisiCalc, MicroSoft BASIC, Atari Word Processor, Letter Perfect Disk (Not ROM), and, I suspect, any program which starts in low memory and looks for RAMTOP to decide the amount of memory available.

Screen Sonics (11416 S. Outer Rd., Chesterfield, MO 63017) is sending me a "Sidewriter", their $250 outboard keyboard for the 400 (or 800) to review for the next issue, so I'll have the most powerful and expensive 400 around!!

Ad Astra, the Atari HAM Newsletter (4749 S.R. 207 N.E., Washington, C.H., OH 43160 $10 yr) in their new issue, #4, announces a new company called Creative Firmware, 707 Auburn Dr., Richardson, TX 75081) which has various new kits, including a OS EPROM Board allowing you to make custom OS and use the extra 4K of the OS not presently used for extending your OS abilities.

Atari has announced a new service contract anyone can get, even if your Atari is out of warranty and even dead!! There are two packages: For the Atari 400 and usual peripherals such as the Cassette Recorder, Paddles, Joysticks, Modem and Interface for $40 a year; and, For the Atari 400 or 800, Disk Drives, printers, RAMs, etc. for $120 a year. These can be expanded to 3 years. A fantastic deal if you have much equipment and good at any Atari service station. Call 1-800-538-8543 for details.

Meteor Storm reviewed by Brian is the first from a local ACE member to become a commercial program. Written by a teenager who taught himself how to program in machine language using De Re Atari, aided by another self-taught teen, it is a fine original game and has been a hit at our meetings.

It is getting close to Christmas, and we will again have a Christmas Disk of all our Best programs since the "Best of 1982" ½ disk, as well as a "Ruth's Pilot" disk by Ruth Ellsworth, a disk full of educational programs for the whole family. There will also be something different: **Alliance Software** has produced a well documented high-speed disk copier which will copy a disk with one drive in two passes and at high speed. This is not a pirate copier, just a fantastic convience. It also makes multiple copies, allows you to see the data being copied and has other features. Alliance has sold ACE a licence to sell copies to our members only. For the month of December only, the above disks are only $10 each. After that, $15 each. Order from Chuck and Jody Ross. They will not be shipped until December so you will recieve them for the holiday season.

Ellsworth: Pilot Listing

```
10 R:TIMED ADDITION ROUTINE
20 R:(C) Ruth Ellsworth
30 R:ACE NEWSLETTER
40 R:3662 VINE MAPLE DR.
50 R:EUGENE, OREGON 97405
60 C:#N=0
70 C:#R=0
80 C:@B1373=16
90 C:@B1374=2
100 WRITE:5,}
110 U:*TITLE
120 E:
130 *TITLE
140 POS:2,5
150 C:@B755=2
160 WRITE:5, TIMED ADDITION
170 T:
180 T:
190 T:PRESS START TO BEGIN
200 R:WAIT UNTIL START IS HIT
210 PA:60
220 J(@B53279<>6):*TITLE2
230 J(@B53279=6):*START
240 *TITLE2
250 POS:2,5
260 C:@B755=4
270 WRITE:5, PROGRAM NAME
280 R:WAIT UNTIL START IS HIT
290 POS:2,5
300 C:@B755=4
310 WRITE:5, TIMED ADDITION
320 R:WAIT UNTIL START IS HIT
330 PA:60
340 J(@B53279<>6):*TITLE
350 J(@B53279=6):*START
360 R:ACE NEWSLETTER
370 *START
371 R:ROUTINE TO COUNT ELAPSED TIME
372 C:#T=@B18*65536+@B19*256+@B20/60[
Internal clocl set - jiffies/60 to gi
ve seconds
380 GR:QUIT
390 T:}
400 U:*ADDITION
410 E:
420 *ADDITION
430 C:#N=#N+1
460 T:}
470 C:#A=?\11
480 POS:10,12
490 T(#A=0):  0
500 T(#A=1):  1  #
510 T(#A=2):  2  $$
520 T(#A=3):  3  %%%
530 T(#A=4):  4  @@@@
540 T(#A=5):  5  .....
550 T(#A=6):  6  ''''''
560 T(#A=7):  7  <<<<<<<
570 T(#A=8):  8  ++++++++
580 T(#A=9):  9  >>>>>>>>>
590 T(#A=10): 10  **********
600 C:#B=?\11
610 POS:10,13
620 T(#B=0):  0
630 T(#B=1):  1  #
640 T(#B=2):  2  $$
650 T(#B=3):  3  %%%
660 T(#B=4):  4  @@@@
670 T(#B=5):  5  .....
680 T(#B=6):  6  ''''''
690 T(#B=7):  7  <<<<<<<
700 T(#B=8):  8  ++++++++
710 T(#B=9):  9  >>>>>>>>>
720 T(#B=10): 10  **********
730 POS:10,13
740 T:+
750 POS:10,14
760 T:____
770 C:#C=#A+#B
780 READ:K,$Y
790 M:#C
800 JY:*RIGHT
810 MS:5
820 JY:*COUNTER
830 T:
840 POS:10,18
850 T:THE ANSWER IS #C  [use inverse
 video for the words: THE ANSWER IS
851 C:#R=#R-1
860 PA:60
870 J:*RIGHT
880 E:
890 *RIGHT
900 POS:10,15
910 T(#C=1):  1  #
920 T(#C=2):  2  $$
930 T(#C=3):  3  %%%
940 T(#C=4):  4  @@@@
950 T(#C=5):  5  .....
960 T(#C=6):  6  ''''''
970 T(#C=7):  7  <<<<<<<
980 T(#C=8):  8  ++++++++
990 T(#C=9):  9  >>>>>>>>>
1000 T(#C=10): 10 **********
1010 T(#C=12): 12 ********** $$
1020 T(#C=11): 11 ********** #
1030 T(#C=13): 13 ********** %%%
1040 T(#C=14): 14 ********** @@@@
1050 T(#C=15): 15 ********** .....
1060 T(#C=16): 16 ********** ''''''
1070 T(#C=17): 17 ********** <<<<<<<
1080 T(#C=18): 18 ********** ++++++++
1090 T(#C=19): 19 ********** >>>>>>>>
>
1100 T(#C=20): 20 ********** ********
**
1110 C:#R=#R+1
1120 PA:60
1130 J:*ADDITION
1140 E:
1150 T:}
1160 POS:10,8
1170 T:THE ANSWER IS #C
1180 J:*RIGHT
1190 *COUNTER
1200 T:}
1210 POS:10,8
1220 T:YOU ANSWERED #R RIGHT
1230 POS:10,10
1231 C:#N=#N-1
1240 T:OUT OF #N PROBLEMS
1250 POS:10,12
1255 C:#X=@B18*65536+@B19*256+@B20/60
1256 C:#G=#X-#T
1260 T(#G<60):IN #G  SECONDS
1270 T(#G>60):IN #G/60  MINUTES
1280 E:
```

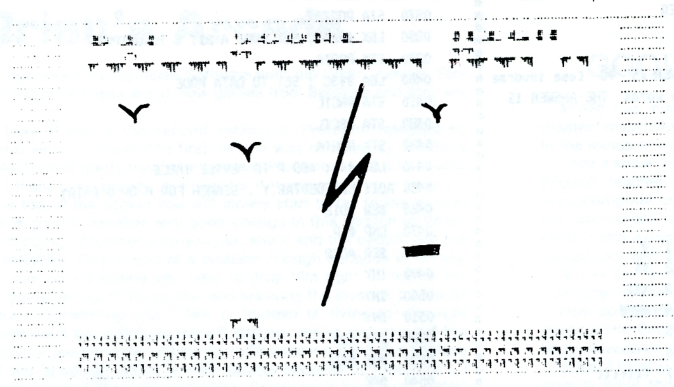
Chastain: printer

```
0100  .OPT OBJ
0110 ;PROGRAM 1 - PRINTER HANDLER
0120 PORTA=$D300 ;PIA - PORT A DATA
0130 PORTB=$D301 ;PIA - PORT B DATA
0140 PACTL=$D302 ;PIA - PORT A CONTROL
0150 PBCTL=$D303 ;PIA - PORT B CONTROL
0160 CR=$9B ; ATARI EOL CHARACTER
0170 ACR=$0D ; ASCII CR
0180 SUCCES=$01 ;CIO SUCCESS STATUS
0190 PTEMP=$1F ;TEMPORARY STORAGE
0200 DEVTAB=$031A ; DEVICE TABLE
0210 PRNBUF=$03C0 ; PRINTER BUFFER
0220  *=$0600
0230 PST .BYTE 0
0240  .BYTE PND-PST+127/128
0250  .WORD PST
0260  .WORD PINIT
0270 ;   *** BOOT CONTINUATION ***
0280  LDA #$3C ; TURN OFF CASSETTE MOTOR
0290  STA PACTL
0300  JMP MOVE
0310  BRK
0320 ;  *** INITIALIZATION ROUTINE ***
0330 PINIT LDA #$38 ; SET TO CONFIGURE PORTS
0340  STA PACTL
0350  STA PBCTL
0360  LDA #$FF ; SET PORT B TO OUTPUT
0370  STA PORTB
0380  LDX #$20 ; SET PORT A BIT 5 TO OUTPUT
0390  STX PORTA
0400  LDA #$3C ; SET TO DATA MODE
0410  STA PACTL
0420  STA PBCTL
0430  STX PORTA
0440  LDY #0 ; ADD P TO DEVICE TABLE
0450 ADI1 LDA DEVTAB,Y ; SEARCH FOR P OR 0 ENTRY
0460  BEQ ADI2
0470  CMP #'P
0480  BEQ ADI2
0490  INY
0500  INY
0510  INY
0520  CPY #30
0530  BNE ADI1
0540  BRK
0550 ADI2 LDA #'P ; PUT P IN DEVICE TABLE
0560  STA DEVTAB,Y
0570  LDA #PST&255 ; PUT HANDLER ENTRY VECTOR INTO DEVICE TABLE
0580  STA DEVTAB+1,Y
0590  LDA #PST/256
0600  STA DEVTAB+2,Y
0610  RTS
0620 ; *** WRITE ROUTINE ***
0630 PHWRIT STA PTEMP ; SAVE CHARACTER FROM CIO
0640 LOOP LDA PORTA ; CHECK FOR ERROR
0650  BMI ERROR ; BRANCH IF ERROR
0660  ASL A ; CHECK FOR BUSY
0670  BPL LOOP ; LOOP IF BUSY
0680  LDA PTEMP ; RESTORE CHARACTER
0690  CMP #CR ; ATARI EOL ($9B) TO ASCII CR ($0D)
0700  BNE CONT1
0710  LDA #ACR
0720 CONT1 EOR #$FF
0730  STA PORTB ; SEND DATA
0740  JSR TIME ; KILL SOME TIME
0750  LDA #0 ; STROBE PRINTER
0760  STA PORTA
0770  JSR TIME ; KILL SOME TIME
0780  LDA #$20
0790  STA PORTA
0800 PHCLOS LDY #SUCCES ; RETURN STATUS TO CIO
0810 TIME RTS
0820 ; *** STATUS ROUTINE ***
0830 PHSTAT LDA PORTA
0840  BPL PHCLOS ; RETURN IF NO ERROR
0850 ERROR LDY #$8A ; RETURN ERROR 138 TO CIO
0860 BADST RTS
0870 ; *** CODE NOT SAVED IN RAM ***
0880 MOVE LDY #14 ; MOVE HANDLER VECTOR TABLE
0890 MOV LDA LOC,Y
0900  STA PST,Y
0910  DEY
0920  BPL MOV
0930  CLC
0940  RTS
0950 ; *** HANDLER VECTOR TABLE ***
0960 LOC .WORD PHSTAT-1 ;OPEN ADDRESS
0970  .WORD PHCLOS-1 ;CLOSE ADDRESS
0980  .WORD BADST-i ;READ
0990  .WORD PHWRIT-1 ;WRITE ADDRESS
1000  .WORD PHSTAT-1 ;STATUS ADDRESS
1010  .WORD BADST-1 ;SPECIAL
1020  JMP BADST
1030 PND .END
```

Chastain Listing 2

```
0100  .OPT OBJ
0110 ;PROGRAM 2 - CREATES CASSETTE BOOTABLE T
APE.
0120 ;ADAPTED FROM ATARI USER'S MANUAL.
0130 ;
0140 ;GIVEN:
0150 PST = $0600 ;PROGRAM START ADDRESS
0160 PND = $069D ;PROGRAM END ADDRESS
0170 FLEN= PND-PST+127/128*128
0180 ;
0190 ;ATARI OS EQUATES
0200 OPEN = $3
0210 OPNOT = $8
0220 PUTCHR = $B
0230 CLOSE = $C
```

Go to page 8



VULTURES LISTINGS

```
0 REM *******************
1 REM ** ACE NEWSLETTER **
2 REM **3662 VINE MAPLE **
3 REM **   EUGENE, OR   **
4 REM **     97405      **
5 REM **      $10 YR    **
6 REM **    Nov 1982    **
7 REM *******************
10 REM ******************
20 REM ** VULTURES III  **
30 REM **STAN OCKERS 8/2**
40 REM ******************
90 GRAPHICS 18:POSITION 5,3:? #6;"vultures":POSITI
ON 8,5:? #6;"III"
100 PMHI=1547:IMAGE0=1552:IMAGE1=1553:IMAGE2=1554:
IMAGE3=1555:HPOS3=1559:VPOS3=1567
110 FLAG0=1568:FLAG1=1569:FLAG2=1570
112 COL0=52:COL1=24:COL2=32:COL3=244:BKCOL=144
120 VPOS0=1564:VPOS1=1565:VPOS2=1566:IMGPT=1584
130 HPOS0=1556:HPOS1=1557:HPOS2=1558:RAMTOP=106:PM
BASE=54279:SDMCTL=559:GRACTL=53277
140 PCOLR0=704:PCOLR1=705:PCOLR2=706:PCOLR3=707:PO
KE 1577,10
150 DIM B$(20),BB$(20),V$(2):V$(1)=CHR$(136):V$(2)
=CHR$(138):BB$(1)=CHR$(0):BB$(20)=CHR$(0):BB$(2)=B
B$
151 DIM CL$(38):CL$(1)=" ":CL$(38)=" ":CL$(2)=CL$
152 DIM X$(38),X1$(38):X$(1)="X":X$(38)="X":X$(2)=
X$:X1$(1)="x":X1$(38)="x":X1$(2)=X1$
153 DIM LTNG$(46):RESTORE 154:FOR I=1 TO 46:READ A
:LTNG$(I,I)=CHR$(A):NEXT I
154 DATA 124,29,41,29,30,124,29,30,30,40,29,30,30,
40,29,30,124,29,40,29,124,29,41,29,41,29,41,29,30,
124,29
155 DATA 30,30,41,29,30,30,41,29,30,30,41,29,30,30
,124
156 DIM ERAS$(46):RESTORE 157:FOR I=1 TO 46:READ A
:ERAS$(I,I)=CHR$(A):NEXT I
157 DATA 32,29,32,29,30,32,29,30,30,32,29,30,30,32
,29,30,32,29,32,29,32,29,32,29,32,29,32,29,30,32,2
9
158 DATA 30,30,32,29,30,30,32,29,30,30,32,29,30,30
,32
160 COUNT=20:PERCH=180:BIRDS=0:DIF=3:POKE 1555,8
200 GOSUB 6000:GOSUB 2000:POKE 764,255:? CHR$(28);
"PRESS START TO BEGIN":? "ANY KEY TO PAUSE"
202 ? "Beware the golden vultures !!!":GOSUB 1000
210 A=PEEK(RAMTOP)-16:POKE PMBASE,A:POKE PMHI,A:SO
UND 0,0,0,0:A=USR(1536):GRAPHICS 0:GOSUB 5000
220 POKE SDMCTL,62:POKE GRACTL,3:POKE 764,255
230 COUNT=20:PERCH=180:BIRDS=0:DIF=3:POKE 1555,8:G
OLD=500:SCORE=0
255 SHIELD=20:FOR J=SHIELD TO 23:POSITION 1,J:? X$
;:NEXT J:POSITION 0,0:? " dif  score   high"
260 POKE PCOLR0,243:POKE PCOLR1,243:POKE PCOLR2,24
3:POKE PCOLR3,40
270 POKE 752,1:FOR I=2 TO 36 STEP 2:POSITION I,1:?
 V$:NEXT I:BLEFT=18:BLAND=0:POKE 1575,0:HIT=0
275 POSITION 1,SHIELD-1:? CL$
277 POSITION 22,0:? DIF:POSITION 26,0:? SCORE:POSI
TION 34,0:? HIGH
280 RESTORE 285:FOR I=1564 TO 1571:READ A:POKE I,A
:NEXT I:POKE 1559,100:POKE 1783,11
285 DATA 0,0,0,150,0,0,0,0
290 B$=BB$
299 POKE 712,BKCOL:POKE 1574,0:POKE 708,COL0:POKE
709,COL1:POKE 710,COL2:POKE 711,COL3
300 COUNT=COUNT-1:A=PEEK(1575):IF HIT<A THEN HIT=A
:POKE 1788,129:SCORE=SCORE+10*DIF:POSITION 26,0:?
SCORE
310 IF PEEK(FLAG0)=0 AND COUNT<0 THEN GOSUB 905:PO
KE (FLAG0),1:POKE (HPOS0),40+R*8:POKE VPOS0,48
320 IF PEEK(FLAG0)=1 THEN POKE VPOS0,PEEK(VPOS0)+D
IF
330 IF PEEK(VPOS0)>=PERCH THEN POKE FLAG0,0:A=INT(
(PEEK(HPOS0)-48)/4):GOSUB 950:POKE HPOS0,0:POKE VP
OS0,32
340 IF PEEK(FLAG1)=0 AND COUNT<0 THEN GOSUB 905:PO
KE (FLAG1),1:POKE (HPOS1),40+R*8:POKE VPOS1,48
350 IF PEEK(FLAG1)=1 THEN POKE VPOS1,PEEK(VPOS1)+D
IF
360 IF PEEK(VPOS1)>=PERCH THEN POKE FLAG1,0:A=INT(
(PEEK(HPOS1)-48)/4):GOSUB 950:POKE HPOS1,0:POKE VP
OS1,32
370 IF PEEK(FLAG2)=0 AND COUNT<0 THEN GOSUB 905:PO
KE (FLAG2),1:POKE (HPOS2),40+R*8:POKE VPOS2,48
380 IF PEEK(FLAG2)=1 THEN POKE VPOS2,PEEK(VPOS2)+D
IF
390 IF PEEK(VPOS2)>=PERCH THEN POKE FLAG2,0:A=INT(
(PEEK(HPOS2)-48)/4):GOSUB 950:POKE HPOS2,0:POKE VP
OS2,32
392 IF BLAND>2 THEN 500
394 IF BLEFT<1 THEN 600
400 FLASH=FLASH-1:IF FLASH<0 THEN GOTO 550
405 IF PEEK(1574)>0 THEN GOTO 650
410 GOLD=GOLD-1:IF GOLD<0 AND BLAND>3 THEN GOLD=10
0*(9-DIF):DIF=DIF+2:XCNT=30:POKE 704,44:POKE 705,4
4:POKE 706,44
420 IF XCNT>0 THEN XCNT=XCNT-1:IF XCNT=0 THEN DIF=
DIF-2:POKE 704,243:POKE 705,243:POKE 706,243:POSIT
ION 22,0:? DIF
425 IF PEEK(764)<255 THEN GOSUB 1000
430 GOTO 300
499 REM ** REMOVE A SHIELD **
500 POKE 1789,207:SVCOL=COL1:FOR I=1 TO 6:POKE 708
,PEEK(708)+2:POSITION 1,SHIELD:? X1$:FOR J=1 TO 5:
NEXT J
502 POSITION 1,SHIELD:? X$:FOR J=1 TO 5:NEXT J:NEX
T I:POSITION 1,SHIELD-1:? CL$:POSITION 1,SHIELD:?
CL$
508 SHIELD=SHIELD+1:PERCH=PERCH+8:IF SHIELD=23 THE
N 700
510 DIF=DIF-1:IF DIF<2 THEN DIF=2
520 GOTO 270
550 POKE 1789,171:POKE 712,40:X=RND(0)*23+8:POSITI
ON X,2:? LTNG$;:POKE 712,15:POSITION X,2:? ERAS$;:
POKE 712,BKCOL
555 A=9-DIF:FLASH=RND(0)*30*A+10*A
560 GOTO 405
599 REM ** REMAINING BIRDS LEAVE **
600 DIF=DIF+1:IF DIF>6 THEN DIF=6
612 FOR I=1 TO 150
614 P=PEEK(VPOS0):IF P>0 THEN POKE VPOS0,P-1
616 P=PEEK(VPOS1):IF P>0 THEN POKE VPOS1,P-1
618 P=PEEK(VPOS2):IF P>0 THEN POKE VPOS2,P-1
620 NEXT I:POKE 77,0:GOTO 270
649 REM ** HIT BY LIGHTNING **
650 POKE HPOS3,0:POKE 53278,0:POKE 1789,189:POKE 7
07,BKCOL
655 FOR I=8 TO 48 STEP 4:POKE 712,I:FOR J=1 TO 30:
NEXT J:NEXT I:POKE 712,BKCOL
660 FOR I=1 TO 200:NEXT I:POKE 1788,79:FOR J=1 TO
DIF
665 SCORE=SCORE-100:IF SCORE<0 THEN SCORE=0
670 POSITION 26,0:? SCORE;"   ":FOR L=1 TO 100:NEX
T L:NEXT J
690 POKE 707,40:GOTO 270
700 IF SCORE>HIGH THEN HIGH=SCORE
705 POKE GRACTL,0:FOR I=53261 TO 53264:POKE I,0:NE
XT I
710 GRAPHICS 18:POSITION 5,3:? #6;"final score "
720 POSITION 8,5:? #6;SCORE
730 POSITION 5,8:? #6;"PRESS START":POSITION 4,9:?
 #6;"TO PLAY AGAIN"
740 IF PEEK(53279)<>6 THEN 740
750 GRAPHICS 0:GOSUB 5000:GOTO 220
900 REM ** START ANOTHER BIRD **
905 COUNT=INT(RND(0)*30)
910 R=INT(RND(0)*18)+2:IF ASC(B$(R))>0 THEN 910
920 B$(R,R)=CHR$(1):POSITION 2*(R-1),1:? "  ":BLEF
T=BLEFT-1:RETURN
949 REM ** BIRD AT SHIELD **
950 POSITION A,(PERCH-32)/8:? V$;:POKE 1788,150:BL
AND=BLAND+1:RETURN
999 REM ** PAUSE ROUTINE **
1000 POKE 764,255
1010 IF PEEK(53279)<>6 THEN 1010
1020 RETURN
1999 REM ** VBI ROUTINE **
2000 RESTORE 2001:DIM VB$(348):FOR I=1 TO 348:READ
 A:VB$(I,I)=CHR$(A):NEXT I
2001 DATA 72,138,72,152,72
2002 DATA 206,40,6,173,40,6,16,29,173,41,6,141,40,
6,162,2,254,16,6,254,16,6,189,16,6,201,8,144,5,169
,0,157,16,6
2004 DATA 202,16,235
2005 DATA 174,120,2,224,11,208,3,142,247,6,224,7,2
08,3,142,247,6
2006 DATA 174,247,6,173,23,6,201,200,176,10,224,7,
208,6,238,23,6,238,23,6,201,48,144,10,224,11,208,6
2008 DATA 206,23,6,206,23,6
2010 DATA 24,173,11,6,105,4,133,204,162,0,134,207,
160,0,132,203,189,20,6,157,0,208,189,12,6,221,16,6
,208,8,189,28
2020 DATA 6,221,24,6,240,69,189,16,6,157,12,6,189,
28,6,157,24,6,165,203,221,28,6,240,10,169,0,145,20
3,230,203,240
2030 DATA 42,208,239,189,16,6,170,189,48,6,133,205
,189,49,6,133,206,177,205,240,14,145,203,230,205,2
08,2
2040 DATA 230,206,230,203,240,10,208,238,169,0,145
,203,230,203,208,250,230,204,166,207,232,134,207,2
24,4,144,154
2042 DATA 173,15,208,240,34,106,144,2,162,0,106,14
4,2,162,1,106,144,2,162,2,169,0,157,28,6,157,32,6
2044 DATA 141,30,208,238,39,6,169,2,141,132,6
2046 DATA 173,7,208,240,3,141,38,6
2050 DATA 174,252,6,240,36,206,248,6,16,31,189,0,6
,141,0,210,232,189,0,6,141,1,210
2052 DATA 232,189,0,6,240,9,141,248,6,232,142,252,
6,208,3,141,252,6
2054 DATA 174,253,6,240,36,206,249,6,16,31,189,0,6
,141,2,210,232,189,0,6,141,3,210
2056 DATA 232,189,0,6,240,9,141,249,6,232,142,253,
6,208,3,141,253,6
2080 DATA 104,168,104,170,104,76,98,228
2090 GOSUB 3010:GOSUB 4100:RETURN
3000 REM * PAGE 6 - INSERT VBI ROUTINE *
3010 RESTORE 3020:FOR I=1536 TO 1545:READ A:POKE I
,A:NEXT I
3020 DATA 104,160,0,162,0,169,7,76,92,228
3030 A=ADR(VB$):B=INT(A/256):C=A-256*B:POKE 1538,C
:POKE 1540,B:RETURN
4099 REM ** SOUNDS **
4100 RESTORE 4120:FOR I=1665 TO 1766:READ A:POKE I
,A:NEXT I
4120 DATA 60,170,3,54,170,3,48,170,3,45,170,3,40,1
70,3,36,170,3,0,0,0
4132 DATA 10,204,3,11,204,3,10,204,3,9,204,3,8,204
,3,7,204,3,0,0,0
4142 DATA 20,143,1,80,143,3,90,140,6,95,137,20,100
,134,30,0,0,0
4144 DATA 13,12,20,16,143,30,12,8,40,16,132,60,23,
130,70,0,0,0
```

Go to page 7

Ockers Vulture Listing cont

```
4146 DATA 40,174,20,50,172,20,60,170,20,70,168,20,
80,166,20,90,164,20,105,162,20,0,0,0
4200 REM ** IMAGES **
4210 DIM V0$(4):RESTORE 4220:FOR I=1 TO 4:READ A:V
0$(I)=CHR$(A):NEXT I
4215 V0=ADR(V0$):POKE IMGPT+1,INT(V0/256):POKE IMG
PT,V0-256*PEEK(IMGPT+1)
4220 DATA 36,90,153,0
4230 DIM V1$(5):RESTORE 4240:FOR I=1 TO 5:READ A:V
1$(I)=CHR$(A):NEXT I
4235 V1=ADR(V1$):POKE IMGPT+3,INT(V1/256):POKE IMG
PT+2,V1-256*PEEK(IMGPT+3)
4240 DATA 66,165,24,24,0
4250 DIM V2$(5):RESTORE 4260:FOR I=1 TO 5:READ A:V
2$(I)=CHR$(A):NEXT I
4255 V2=ADR(V2$):POKE IMGPT+5,INT(V2/256):POKE IMG
PT+4,V2-256*PEEK(IMGPT+5)
4260 DATA 195,36,24,24,0
4270 DIM V3$(6):RESTORE 4280:FOR I=1 TO 6:READ A:V
3$(I)=CHR$(A):NEXT I
4275 V3=ADR(V3$):POKE IMGPT+7,INT(V3/256):POKE IMG
PT+6,V3-256*PEEK(IMGPT+7)
4280 DATA 129,66,36,24,24,0
4290 DIM V4$(6):RESTORE 4296:FOR I=1 TO 6:READ A:V
4$(I)=CHR$(A):NEXT I
4292 V4=ADR(V4$):POKE IMGPT+9,INT(V4/256):POKE IMG
PT+8,V4-256*PEEK(IMGPT+9)
4296 DATA 255,255,255,255,255,0
4300 RETURN
4999 REM ** CHANGE DISPLAY LIST **
5000 DL=PEEK(560)+256*PEEK(561)
5010 POKE DL+3,70:POKE DL+6,6:FOR I=DL+7 TO DL+28:
POKE I,4:NEXT I:RETURN
6000 GRAPHICS 0:POKE 752,1:POSITION 8,1:? "***  VU
LTURES III  ***"
6010 POSITION 5,3:? "The vultures are landing and
":? "removing protective layers above"
6020 ? "you!  There are three layers and ":? "ever
y time three vultures land":? "one layer will disa
ppear.":?
6030 ? "  You can stop the birds with":? "a remov
al device controlled by":? "joystick 0. You get te
n times"
6040 ? "the difficulty for each bird you":? "stop.
 When you have removed a"
6050 ? "flock, the difficulty will go up.":? :? "I
f you get hit by lightning your"
6060 ? "score will decrease by a hundred":? "times
 the difficulty.":? :? "Just a sec..."
6090 RETURN
```

SOUNDS LISTING

```
0 REM ********************
1 REM ** ACE NEWSLETTER **
2 REM **3662 VINE MAPLE **
3 REM **   EUGENE, OR   **
4 REM **      97405     **
5 REM **     $10 YR     **
6 REM **    Nov 1982    **
7 REM ********************
8 REM *SOUNDS UTILITY BY*
9 REM * STAN OCKERS 8/2 *
100 GOSUB 5000:GOSUB 3000:A=USR(1536):SOUND 0,0,0,
0
110 DIM A$(3),P$(40),D$(40),L$(40),T$(40),Z$(40),N
$(40),V$(4)
120 Z$(1)=CHR$(0):Z$(40)=CHR$(0):Z$(2)=Z$:RESTORE
130:FOR I=1 TO 4:READ A:V$(I,I)=CHR$(A):NEXT I:VN=
1:P=1
130 DATA 2,7,12,17
280 P$=Z$:D$=Z$:L$=Z$:T$=Z$
290 GRAPHICS 0:POKE 703,4:POKE 752,1:POSITION 2,0:
? #6;" ":POSITION 3,4:? #6;"PITCH":POSITION 3,9:?
#6;"DURATION"
292 POSITION 3,14:? #6;"LOUDNESS":POSITION 3,19:?
#6;"DISTORTION"
300 V=ASC(V$(VN)):POSITION P,V+1:? #6;" ";:S=STICK
(0):IF STRIG(0)=0 THEN 350
310 IF (S=6 OR S=10 OR S=14) AND VN>1 THEN VN=VN-1
320 IF (S=5 OR S=9 OR S=13) AND VN<4 THEN VN=VN+1
330 IF (S=5 OR S=6 OR S=7) AND P<38 THEN P=P+1
340 IF (S=9 OR S=10 OR S=11) AND P>1 THEN P=P-1
350 V=ASC(V$(VN)):POSITION P,V+1:? #6;"^";
360 IF VN=1 THEN A=ASC(P$(P)):MAX=254:DEL=2
370 IF VN=2 THEN A=ASC(D$(P)):MAX=127:DEL=1
380 IF VN=3 THEN A=ASC(L$(P)):MAX=15:DEL=1
390 IF VN=4 THEN A=ASC(T$(P)):MAX=14:DEL=2
400 IF STRIG(0)=1 THEN 425
410 IF (S=14 OR S=6 OR S=10) AND A<MAX THEN A=A+DE
L
420 IF (S=13 OR S=5 OR S=9) AND A>0 THEN A=A-DEL
425 IF PEEK(53279)=6 THEN GOSUB 4000:GOTO 490
430 IF VN=1 THEN P$(P,P)=CHR$(A)
440 IF VN=2 THEN D$(P,P)=CHR$(A)
450 IF VN=3 THEN L$(P,P)=CHR$(A)
460 IF VN=4 THEN T$(P,P)=CHR$(A)
470 GOSUB 6000
490 GOTO 300
3000 REM * PAGE 6 - INSERT VBI ROUTINE *
3010 RESTORE 3020:FOR I=1536 TO 1545:READ A:POKE I
,A:NEXT I
3020 DATA 104,160,0,162,0,169,7,76,92,228
3030 A=ADR(VB$):B=INT(A/256):C=A-256*B:POKE 1538,C
:POKE 1540,B:RETURN
4000 SAV=P:P=0:PS=1536:? CHR$(125)
4010 P=P+1:PS=PS+1
4020 PIT=ASC(P$(P))/2:LD=ASC(L$(P)):DUR=ASC(D$(P))
:IF DUR=0 THEN 4100
4030 POKE PS,PIT:PS=PS+1:? PIT;",";
4040 DIS=INT(ASC(T$(P))):CTL=16*DIS+LD:IF CTL=0 TH
EN CTL=LSTCTL
4050 LSTCTL=CTL:POKE PS,CTL:? CTL;",";:PS=PS+1
4070 POKE PS,DUR:? DUR;",";:GOTO 4010
4100 FOR I=0 TO 2:POKE PS+I,0:? "0,";:NEXT I:POKE
1788,1
4110 P=SAV:RETURN
5000 RESTORE 5010:DIM VB$(44):FOR I=1 TO 44:READ A
:VB$(I,I)=CHR$(A):NEXT I:RETURN
5010 DATA 174,252,6,240,36,206,248,6,16,31,189,0,6
,141,0,210,232,189,0,6,141,1,210
5012 DATA 232,189,0,6,240,9,141,248,6,232,142,252,
6,208,3,141,252,6
5014 DATA 76,98,228
6000 A$=STR$(A):L=LEN(A$):FOR I=0 TO 2:IF I<L THEN
 POSITION P,V-I:? #6;A$(L-I,L-I);
6010 IF I>=L THEN POSITION P,V-I:? #6;" ";
6020 NEXT I:RETURN
```

```
  10 ; SOUND IN VBI ROUTINE
  20 AUDC1    =    $D201      ;(53761)
  30 AUDF1    =    $D200      ;(53760)
  40 DURCNT   =    $6F8       ;(1784)
  50 STRPOS   =    $6FC       ;(1788)
  60 AREA     =    $600       ;(1536)
  70          *=   $0000
  80 TUNE1    LDX  STRPOS     ; Position is index
  90          BEQ  TUNE2      ; No sound
  0100        DEC  DURCNT     ; Wait
  0110        BPL  TUNE2
  0120        LDA  AREA,X     ; Frequency
  0130        STA  AUDF1
  0140        INX
  0150        LDA  AREA,X     ; Control byte
  0160        STA  AUDC1
  0170        INX
  0180        LDA  AREA,X      ; Duration
  0190        BEQ  FIN1        ; Quit if Dur = 0
  0200        STA  DURCNT
  0210        INX
  0220        STX  STRPOS      ; Update string pos.
  0230        BNE  TUNE2       ; Skip over
  0240 FIN1   STA  STRPOS      ; 0 in String pos.
  0250 TUNE2  NOP
```

TINY TEXT REVISITED LISTING

```
0 REM * ACE NEWSLETTER, 3662 VINE MAPLE,EUGENE, OR
 97405 $10 YEAR
1 REM **** TINY TEXT ****
2 REM
3 REM Stan Ockers Sept-81
4 REM ACE Newsletter Nov-81
5 REM
6 REM Mod by Jim Carr 01-OCT-82
7 REM
12 DIM SP$(40):FOR I=1 TO 40:SP$(I,I)=" ":NEXT I
14 DIM S$(45),I$(120),A$(128):SIZ=FRE(0)-50:DIM T$
(SIZ):FOR I=1 TO 45:READ A:S$(I)=CHR$(A):NEXT I
20 DATA 104,104,133,204,104,133,203,104,133,206,10
4,133,205,104,104,168,162,0,161,203,145,203,198,20
3,165
30 DATA 203,201,255,208,2,198,204,165,203,197,205,
208,236,165,204,197,206,208,230,96
40 FOR I=1536 TO 1643:READ A:POKE I,A:NEXT I
50 DATA 104,104,133,204,104,133,203,104,133,206,10
4,133,205,162,0,169,240,32,53,6,169,40,32,91,6
60 DATA 165,207,208,8,169,160,32,91,6,24,144,10,16
9,40,32,53,6,169,120,32,91,6,169,240,32,53,6,96
70 DATA 133,208,161,203,201,96,176,11,201,32,176,5
,24,105,64,208,2,233,32,129,205,230,203,208,2
80 DATA 230,204,230,205,208,2,230,206,198,208,208,
221,96,133,208,169,0,129,205,230,205,208,2
90 DATA 230,206,198,208,208,244,96
110 P=241:POKE 207,0:POKE 82,0:OPEN #2,4,0,"E:":T$
(1)=".":T$(480)=".":T$(2)=T$
120 SCR=PEEK(88)+256*PEEK(89)+120:LL=35:LM=1:IND=5
:TAB=10:PS=66:FF=6:GOTO 500
290 ? "INSERT TEXT OR ...  PRESS SELECT TO EDIT":R
EM SELECT IN INVERSE
300 POSITION 0,0:? SIZ-LEN(T$);" FREE    ":S=STICK(
0):IF S=15 THEN 330
305 IF S=14 AND P<LEN(T$)-320 THEN P=P+40
310 IF S=13 AND P>280 THEN P=P-40
315 IF S=11 AND P<LEN(T$)-280 THEN P=P+1
320 IF S=7 AND P>241 THEN P=P-1
330 A=USR(1536,ADR(T$)+P-241,SCR)
335 K=0
340 POKE 53279,8:PK=PEEK(53279):IF PK=5 THEN GOSUB
 900
350 IF PK=3 THEN 500
360 IF PEEK(764)<255 THEN 400
365 K=K+1:IF K<10 THEN 340
370 IF STRIG(0)=0 THEN P=LEN(T$)-240:POKE 207,0
380 GOTO 300
400 POSITION 0,10:INPUT #2;I$:PK=PEEK(207):IF PK=0
 THEN A$=""
405 LI=LEN(I$):LT=LEN(T$):IF LI=0 THEN 460
407 IF LI+LT>SIZ THEN POSITION 0,1:? "OUT OF SPACE
      ":GOTO 300
410 IF PK=1 THEN A$=T$(P,P+39):IF T$(P+39,P+39)="
" THEN I$(LI+1)=" ":LI=LI+1
420 LA=LEN(A$):AD=ADR(T$):IF LI>LA THEN A=USR(ADR(
S$),AD+LT-1,AD+P-2,LI-LA)
430 T$(P,P+LI-1)=I$
440 IF LA>LI THEN T$(P+LI)=T$(P+LA)
450 P=P+LI:T$(LT+LI-LA+1)="":POKE 207,0:GOTO 300
460 IF PEEK(207)=1 THEN 470
465 IF P<LEN(T$)-279 THEN T$(P)=T$(P+40)
470 POKE 764,255:GOTO 300
500 TRAP 950:ST=PEEK(560)+PEEK(561)*256+4:POKE ST-
1,70:POKE ST+2,7:POKE ST+3,112:POKE ST+4,6:POKE ST
+5,6:POKE ST+24,65
510 POKE ST+25,PEEK(560):POKE ST+26,PEEK(561)
515 OP=OP+1:IF OP=6 THEN OP=1
520 ? CHR$(125):POSITION 20,0:IF OP=1 THEN ? "LOAD
"
522 IF OP=2 THEN ? "EDIT"
534 IF OP=3 THEN ? "PRINT"
536 IF OP=4 THEN ? "SAVE"
538 IF OP=5 THEN ? "DISPLAY"
540 POSITION 0,1:? "PRESS START TO BEGIN"
550 FOR D=1 TO 30:NEXT D
555 POKE 53279,8:IF PEEK(53279)=3 THEN 515
557 IF PEEK(53279)<>6 THEN 555
560 POKE 764,255:POSITION 20,1:? CHR$(125):POSITIO
N 0,1:ON OP GOTO 2000,290,590,1500,590
580 REM **     █= ESC TAB, ARROWS FOLLOWING LINE  I
N INVERSE **
590 FOR I=1 TO 6:? "█";:NEXT I:? :FOR I=1
 TO 6:? "       ";:NEXT I
594 POSITION 0,1:? "SET FORMAT CONTROLS":POSITION
0,6:REM NEXT IN INVERSE:      ? "LINE LEFT IN-  TAB
  PAGE FORM"
```

Go to page 8

Chastain Listing 2 cont

```
0240 CR = $9B
0250 ICCOM = $342
0260 ICBAL = $344
0270 ICBAH = $345
0280 ICBLL = $348
0290 ICBLH = $349
0300 ICAX1 = $34A
0310 ICAX2 = $34B
0320 CIOV = $E456
0330 ;
0340 ;PROGRAM'S ORIGIN
0350  *= $4000
0360 BOOTB LDX #$20
0370  LDA #OPEN
0380  STA ICCOM,X
0390  LDA #OPNOT
0400  STA ICAX1,X
0410  LDA #$80
0420  STA ICAX2,X
0430  LDA #CFILE&$FF
0440  STA ICBAL,X
0450  LDA #CFILE/$100
0460  STA ICBAH,X
0470  JSR CIOV
0480  BMI CERR
0490  LDA #PUTCHR
0500  STA ICCOM,X
0510  LDA #PST&$FF
0520  STA ICBAL,X
0530  LDA #PST/$100
0540  STA ICBAH,X
0550  LDA #FLEN&$FF
0560  STA ICBLL,X
0570  LDA #FLEN/$100
0580  STA ICBLH,X
0590  JSR CIOV
0600  BMI CERR
0610  LDA #CLOSE
0620  STA ICCOM,X
0630  JSR CIOV
0640  BRK
0650 CERR BRK
0660 CFILE .BYTE "C:",CR
0670  .END
```

Chastain Listing 3

```
0100 ;PROGRAM 3 - PRINTER HANDLER DISK VERSION
0110 ; MAKE THIS AN AUTORUN.SYS FILE
0120 DOSINI=$0C ;DISK INITIALIZATION VECTOR
0130 PORTA=$D300 ;PIA - PORT A DATA
0140 PORTB=$D301 ;PIA - PORT B DATA
0150 PACTL=$D302 ;PIA - PORT A CONTROL
0160 PBCTL=$D303 ;PIA - PORT B CONTROL
0170 CR=$9B ; ATARI EOL CHARACTER
0180 ACR=$0D ; ASCII CR
0190 SUCCES=$01 ;CIO SUCCESS STATUS
0200 PTEMP=$1F ;TEMPORARY STORAGE
0210 DEVTAB=$031A ; DEVICE TABLE
0220 PRNBUF=$03C0 ; PRINTER BUFFER
0230  *=$0600
0240 ; *** HANDLER VECTOR TABLE ***
0250 PST .WORD PHSTAT-1 ;OPEN ADDRESS
0260  .WORD PHCLOS-1 ;CLOSE ADDRESS
0270  .WORD BADST-1 ;READ
0280  .WORD PHWRIT-1 ;WRITE ADDRESS
0290  .WORD PHSTAT-1 ;STATUS ADDRESS
0300  .WORD BADST-1 ;SPECIAL
0310  JMP BADST
0320 ; *** PUT NEW "P" IN DEVICE TABLE ***
0330 PINIT JSR PINIT ;PINIT REPLACED WITH DOSINI VEC
TOR
0340 INI LDA #$38 ; SET TO CONFIGURE PORTS
0350  STA PACTL
0360  STA PBCTL
0370  LDA #$FF ; SET PORT B TO OUTPUT
0380  STA PORTB
0390  LDX #$20 ; SET PORT A BIT 5 TO OUTPUT
0400  STX PORTA
0410  LDA #$3C ; SET TO DATA MODE
0420  STA PACTL
0430  STA PBCTL
0440  STX PORTA
0450  LDY #0 ; ADD P TO DEVICE TABLE
0460 ADI1 LDA DEVTAB,Y ; SEARCH FOR P OR 0 ENTRY
0470  BEQ ADI2
0480  CMP #'P
0490  BEQ ADI2
0500  INY
0510  INY
0520  INY
0530  CPY #30
0540  BNE ADI1
0550  BRK
0560 ADI2 LDA #'P ; PUT P IN DEVICE TABLE
0570  STA DEVTAB,Y
0580  LDA #PST&255 ; PUT HANDLER ENTRY VECTOR INTO D
EVICE TABLE
0590  STA DEVTAB+1,Y
0600  LDA #PST/256
0610  STA DEVTAB+2,Y
0620  RTS
0630 ; *** WRITE ROUTINE ***
0640 PHWRIT STA PTEMP ; SAVE CHARACTER FROM CIO
0650 LOOP LDA PORTA ; CHECK FOR ERROR
0660  BMI ERROR ; BRANCH IF ERROR
0670  ASL A ; CHECK FOR BUSY
0680  BPL LOOP ; LOOP IF BUSY
0690  LDA PTEMP ; RESTORE CHARACTER
0700  CMP #CR ; ATARI EOL ($9B) TO ASCII CR ($0D)
0710  BNE CONT1
0720  LDA #ACR
0730 CONT1 EOR #$FF
0740  STA PORTB ; SEND DATA
0750  JSR TIME ; KILL SOME TIME
0760  LDA #0 ; STROBE PRINTER
0770  STA PORTA
0780  JSR TIME ; KILL SOME TIME
0790  LDA #$20
0800  STA PORTA
0810 PHCLOS LDY #SUCCES ; RETURN STATUS TO CIO
0820 TIME RTS
0830 ; *** STATUS ROUTINE ***
0840 PHSTAT LDA PORTA
0850  BPL PHCLOS ; RETURN IF NO ERROR
0860 ERROR LDY #$8A ; RETURN ERROR 138 TO CIO
0870 BADST RTS
0880 ; SAVE DOSINI VECTOR, SET NEW DOSINI VALUE
0890 DOSSAV LDA DOSINI ; SAVE DOSINI
0900  STA PINIT+1
0910  LDA DOSINI+1
0920  STA PINIT+2
0930  LDA #PINIT&255 ;SET DOSINI
0940  STA DOSINI
0950  LDA #PINIT/256
0960  STA DOSINI+1
0970  JSR INI ; INITIALIZE HANDLER
0980  RTS
0990  *= $2E2
1000  .WORD DOSSAV ; SET RUN ADDRESS
1010  .END
```

☆☆

TinyText Revisited cont

```
595 REM INVERSE:? "SIZE MARG DENT STOP SIZE FEED":
? "█";LL;",█";LM;",█";IND;",█";TAB;",█";PS;",█";FF
:POSITION 0,8
600 INPUT LL,LM,IND,TAB,PS,FF:P=240:P=240:GO TO 71
0
610 P=240
670 GOTO 620
710 LINE=0:GRAPHICS 0:POSITION 0,3:FL=0
715 RL=LL:TP=P:B=ASC(T$(TP,TP))
720 RL=LL-IND*(B=9)-TAB*(B=20)
725 IF B=19 AND OP=3 AND LINE<=(PS-FF) THEN LPRINT
 " ":LINE=LINE+1
726 IF B=19 AND OP=5 THEN ?
727 IF B=16 AND OP=3 THEN FOR I=1 TO PS-LINE:LPRIN
T " ":NEXT I:LINE=0
728 IF B=16 AND OP=5 THEN ? :? :? :LINE=0
735 C=0:K=0
740 K=K+1:TP=TP+1:IF K=RL+1 THEN 765
745 IF TP>LEN(T$)-241 THEN FL=1:GOTO 810
750 A=ASC(T$(TP,TP)):IF A<32 THEN C=0:GOTO 780
755 IF A=32 THEN C=C+1
760 GOTO 740
765 IF C=0 THEN A$=T$(P+1,TP-1):TP=TP-1:GOTO 810
767 IF T$(TP,TP)=" " THEN A$=T$(P+1,TP-1):GOTO 810
768 IF T$(TP-1,TP-1)=" " THEN C=C-1
770 K=1
775 TP=TP-1:IF T$(TP,TP)<>" " THEN K=K+1:GOTO 775
780 IF TP=P+1 THEN P=TP:GOTO 715
785 A$="":I=P+1
790 A$(LEN(A$)+1)=T$(I,I):IF T$(I,I)<>" " THEN 805
795 IF C>1 THEN A=INT(K/C+RND(0)):IF A>0 THEN FOR
J=1 TO A:A$(LEN(A$)+1)=" ":NEXT J:K=K-A
800 C=C-1
802 IF C=1 AND K>0 THEN FOR J=1 TO K:A$(LEN(A$)+1)
=" ":NEXT J
805 I=I+1:IF I<TP THEN 790
810 IF FL THEN A$=T$(P+1,TP-1)
815 IF OP=3 THEN LINE=LINE+1:IF LINE>(PS-FF) THEN
LINE=1:FOR I=1 TO FF:LPRINT " ":NEXT I
820 SP=LM+(B=9)*IND+(B=20)*TAB+(B=3)*(LL-LEN(A$))/
2:IF SP>40 THEN SP=40
830 IF OP=3 THEN LPRINT SP$(1,SP);A$
840 IF OP=5 THEN ? SP$(1,SP);A$
850 IF FL THEN 500
860 P=TP:GOTO 715
900 PK=PEEK(207):IF PK=1 THEN POKE 207,0:GOTO 930
910 IF PK=0 AND P<LEN(T$)-279 THEN POKE 207,1
930 A=USR(1536,ADR(T$)+P-241,SCR):FOR D=1 TO 50:NE
XT D:RETURN
950 ? "ERROR ";PEEK(195);" AT ";256*PEEK(187)+PEEK
(186):? "PRESS RETURN TO CONTINUE":INPUT I$:GOTO 5
00
1500 ? " ENTER FILE NAME":INPUT I$:OPEN #3,8,0,I$:
N=INT(LEN(T$)/128):PRINT #3;N:IF N=0 THEN ST=0:GOT
O 1520
1510 FOR I=1 TO N:ST=128*I:PRINT #3;T$(ST-127,ST):
NEXT I
1520 PRINT #3;T$(ST+1,LEN(T$)):CLOSE #3:GOTO 500
2000 ? " ENTER FILE NAME":INPUT I$:OPEN #3,4,0,I$:
INPUT #3,N:IF N=0 THEN BEG=-127:GOTO 2020
2010 GRAPHICS 0:FOR I=1 TO N:BEG=128*I-127:INPUT #
3,A$:? A$;:T$(BEG)=A$:NEXT I
2020 INPUT #3,A$:T$(BEG+128)=A$:CLOSE #3:POKE 1536
,104:GOTO 500
```

RENEWAL CODE
The code on your mailing label shows the date you started; eg. 11P1 means you started your membership in November, 1981, and need to send in your renewal in November of 1982. The letter P has to do with the type of membership, P means paid.

ACE Bulletin Board
(503) 343-4352

Our new Bulletin Board System, ACE Armudic, is working great. We have two double-density Percom drives, a Tara 48K Atari 400, and a Hayes Smart-Modem. There are many programs for you to download, including all from the Newsletter this year. We have room for more if you wish to Upload them. Call 24 hours a day. SYSOP now Mike Dunn. No passwords used yet.

HOW TO USE THE ARMUDIC CENTRAL COMMUNICATOR. (availble from Frank Huband, 1206 N. Stafford St., Arlington, VA 22201, $99)

A. Introduction for the Remote User.

To prepare for communication with an ARMUDIC Central Communicator, set your terminal parameters as follows: Translation mode—light; Duplex—full; Send parity—none; Receive parity—mark (ignore); Stop bits—one; and Baud rate—300. After setting these parameters, call the ARMUDIC Communicator and sign on according to its instructions. The Main Menu will then display the options available to you. It is presented the first time in its full descriptive form, with a sentence describing each option. Thereafter, ARMUDIC displays only the initial letters of the Main Menu. Press ×RETURN† or "?"(or any invalid Menu option) to display the full Menu at that time.

After you complete sign-on, just press ×RETURN† (in response to any ARMUDIC request for input) to return to the Main Menu. (Exceptions are specifically noted during operation.) If you don't answer in a minute to an ARMUDIC question, ARMUDIC will say good-by and hang up.

B. User-to-User Messages.

1. ACTIVATE. There may be several user-message files (UMFs) available on ARMUDIC. The name of the currently active UMF is displayed on the same line as the time, preceeding the Main Menu display. To list the names and a brief description of the other user-message files, select Main Menu option "A". Activate any of the listed files by entering the associated number. The remainder of this Section describes the options available for use with each file.

2. SEND. To send a user-to-user message, select Main Menu option "S". ARMUDIC will ask you to enter the addressee of the message, and then offer you two options: 1) to lock the message with a password so only a user knowing the password can read it, and 2) to establish a password with which the message can be deleted. If you select either option, you will then be asked for a three-letter password. In any event, the next requests will be for the message title and text. Indicate the end of the message by two successive ×RETURN†s. ARMUDIC will then display the message to you as it will be sent, and will give you the choice of either sending the message, redoing it, or cancelling it.

3. GET. To read the user-to-user messages, select Main Menu option "G". ARMUDIC will present you with a series of choices. Pressing "S" or "R" in response will display the full headings or full headings and text respectively, of all messages beginning with the most recent entry. Alternatively, entering a digit from 1 to 8 (as indicated) will specify the message elements you want displayed. If you pick this option, you will be given the option of reading forward or backward or picking specific messages (or headings) to read, and then asked which message number to begin with.

You can return to the Main Menu at any time while ARMUDIC is displaying user messages or headings (or other files, as indicated in C1 and E, below) by pressing "Control-C". ARMUDIC will pause if you press "Control-S", and re-start when you press "Control-Q".

4. ERASE. You can erase messages for which you know the erasure password by selecting Main Menu option "E". After you indicate the number of the message you want erased, ARMUDIC will display the time and date the message was sent, and its addressee and sender. You will then be asked to enter the three-letter erasure password established when the message was sent. If you correctly enter the password, the message will then be erased.

C. Sending and Receiving Files on the Download Menu.

1. RECEPTION (Download). To select the Download Menu option, select Main Menu option "D". There may be more than one group of download files; if so, after selecting "D" you will be shown a listing of available download file groups and asked to select one. After making that selection, to then download any listed file, enter the number preceding its name. Any file name preceded by an "*"contains inverse-video characters and can be successfully downloaded only with a terminal set up to receive in "ATASCII" (no-translation) mode. Other files must be downloaded in light-translation mode.

When ARMUDIC is ready to send the selected file, it will send "Hit RETURN to begin download". If you want to store the file, set up your system at this time to save received text. (Note: The ATARI Telelink I cartridge cannot save files to disk or cassette. Use another terminal program if you wish to save downloaded files.) When you have done this, press ×RETURN† and the download process will begin. During download, you can use Control-S, -Q and -C (as described in B.2, above) to pause, restart, and terminate the download. When download is complete, store the file to disk under a convenient name or to a cassette, according to your terminal program's instructions.

To run a downloaded program, sign off ARMUDIC, type **NEW**, and use the BASIC **ENTER** command to get the program into the computer's memory. Error messages may appear as the program **ENTER**s. If the text shown with an error message is not a numbered BASIC program line, the error probably indicates that material was stored on your disk or cassette in addition to the program. BASIC will ignore such material, and so can you. **RUN** the program at this time. If the program **RUN**s successfully, **SAVE** (or **CSAVE**) it to disk or to cassette, using either the original file name or tape (to overwrite the **LIST**ed version of the program) or a new file name or tape (to keep both the **LIST**ed and **SAVE**d versions).

2. TRANSMISSION (Upload). You can upload either programs or other files to make them available on a Download Menu. Since the garbling in transmission of a single character in a **SAVE**d program can make it totally inaccessible, programs to be uploaded should be in **LIST**ed form. To put a program into this form, type **NEW** and then **LOAD** (or **CLOAD**) the program into the computer. Then **LIST** the program to a new cassette or to the disk under a new name. Since many terminal programs do not have the "ATASCII" mode needed to receive inverse-video characters, if at all possible prepare the program so it does not contain inverse-video characters. Place machine language code segments in DATA statements rather than strings, for example.

If your terminal program can upload only from memory, before calling ARMUDIC follow your terminal program's instructions to load from disk or cassette into memory the program or other file you want to upload. After signing onto ARMUDIC, select Main Menu option "U". ARMUDIC will then tell you the maximum file length you can upload, which will be the smaller of its available internal memory and the memory remaining on the Download file disk after allowing for possible user-message file growth. On request, indicate the name (an upper-case letter followed by up to seven numbers or upper case letters) under which you want the file to be listed on the Download Menu, and whether you want the program to be available to everyone, only to those knowing the club password, or only to SYSOP.

To upload a program which contains critical inverse-video text, inform ARMUDIC when asked, and set the terminal program to "ATASCII" (no-translation) mode when you are instructed. In any event, when ARMUDIC tells you to begin, start sending your program promptly; if you don't start within one minute ARMUDIC will terminate the upload. After you begin, ARMUDIC will assume a pause of 15 seconds to indicate you have finished, and will begin the process of storing your program to its Download file disk.

If your file was longer than the buffer or disk space available, you will be told, and provided the last few lines of text accepted by ARMUDIC. If more disk space is available, upload the rest of the file in a separate upload. If "Upload save error" is reported, it means ARMUDIC failed in its attempt to save your program — probably because of a bad disk. The message will also be sent to SYSOP's log.

D. Leaving a Message for SYSOP.

To leave a message on SYSOP's printer, if the ARMUDIC system you have called has one, select Main Menu option "L". After selecting "L", enter the message, ending with two ×RETURN†s. This will send the message to SYSOP's printer and return you to the Menu.

E. Other Menu Options.

Menu option "Q" signs you off ARMUDIC. Other Main Menu options may specify information files which ARMUDIC can send to you. During receipt of these files, use Control-S, -Q, and -C (as explained in B2 above) to pause, restart, and terminate receipt of the file.

F. Password Access.

Password holders can gain access to password-protected files and programs by entering "=" in response to a Main Menu command query. You will then be prompted by a "Yes?". Enter the password. ARMUDIC will respond to a correct entry with "O.K.". Any password-protected Main Menu entries, Download Menu files, and user-message files will then be accessible.

G. "Talking" with SYSOP.

SYSOP may interrupt your use of ARMUDIC at the end of any menu selection to communicate with you directly. To respond, merely enter your responses normally through your keyboard, and you will be able to see SYSOP's comments on your screen.

NOVEMEBER MEETING

The November meeting of ACE will be held on November 10th at 7:30 PM at the LCC SCIENCE BUILDING, Room 111. Park at the East end of the South parking lot; walk North to the building and use the East enterance.

A New Tutorial Series by Stan Ockers!!

Machine Language Programming

by Stan Ockers

1 : Load and Store

Why machine language? Well, speed mainly. If you're tired of the plodding motion of graphics under Basic, you probably drool at the fast paced action of those all-machine-language games. Is it reasonable to consider writing a game entirely in machine language? No, not for most people anyway, but things can be speeded up considerably by translating selected parts of a Basic program.

Is machine language difficult? Yes, but not because it's that hard to learn. There are only a few types of operations and they are fairly straight forward. The problem comes because we all tend to make simple mistakes. Make a mistake in Basic and you get an error message when you run the program. Make a mistake in machine language and the Atari "locks up"; no error messages or anything ... nothing to do but power down and re-try after entering the program with appropriate changes.

The brains of the Atari is the 6502 CPU, (Central Processing Unit). It understands only about 70 different instructions contained in the 6502 INSTRUCTION SET. The CPU works its way sequentially through memory looking for the different operation codes, (OP CODES), which tell it the operation to perform. The op code takes one byte or one memory location. An additional one or two bytes may be needed to specify an address to go along with the operation. A common mistake causing programs to "blow up" is not having the proper number of bytes following the op code; two instead of one for example. The CPU then interprets part of what was supposed to be an address as the next operation and dutifully tries to execute it. The result is the computer going off to "never-never land".

It's most convenient to express numbers representing op codes etc. in the hexidecimal, (HEX), numbering system. In Hex a digit can have one of sixteen values, (0-9 or A-F). Two hex digits can hold 256 different values ($00-$FF). Each memory location in the Atari can also have one of 256 different values. Hex values are usually preceded by a dollar sign to indicate they are hex.

Memory is a place for storing numbers. It is important to distinguish between the ADDRESS of a memory location and the DATA or number contained at that address. The 6502 can directly address over 65,000 different locations, (65K bytes). It takes four hex digits to specify an address and they run from $0000 to $FFFF. The two leftmost hex digits specify the PAGE. Page 1 for example contains all addresses between $0100 and $01FF, (256 bytes). A number at an address may represent an op code or additional information needed in an operation.

Some addresses have a direct relationship to actual physical devices in the Atari. In graphics 0 there are 960 addresses which hold the data appearing as characters on the screen. Change the values in these locations and you change what shows up on the screen. Where are these locations? Well, it varies depending on the size of your memory, graphics mode, etc. Two locations in page zero contain the first address (upper left hand corner of the screen). $0058 holds the LOW ORDER BYTE or two rightmost digits of the hex address while $0059 holds the HIGH ORDER BYTE or page of the address. Address order is normally reversed like this; low order byte first then the high order byte.

The 6502 CPU has provision for holding numbers temporarily but unlike memory locations these locations have no address. They are called REGISTERS. The op codes specify how numbers are to be transferred among these registers or between registers and memory. The most often used register is called the ACCUMULATOR. The most often used instructions involve loading the accumulator with a number or storing the number from the accumulator somewhere else. Three letter code words called MNEMONICS specifying these operations are LDA (load the accumulator) and STA (store the accumulator). We will use these instructions to write a machine language program.

The sound chip in the Atari, POKEY, also has some registers but since they are outside the CPU they have addresses. The number in $D200 (AUDF1) controls the pitch of voice 1 while the number in $D201 (AUDC1) controls the distortion and loudness of voice 1. A program to produce middle C might look like this in words:

Start Load the accumulator with the desired pitch
Store the accumulator in $D200
Load the accumulator with a number representing distortion and volume
Store the accumulator in $D201
End

In loading or storing the accumulator the location for data must somehow be specified. The type of specification is decided by the ADDRESSING MODE. The op code itself specifies the addressing mode. Table 1 is a very abbreviated op code table giving the op codes for three different addressing modes of STA and LDA, immediate, zero page and absolute.

A program resides in a sequence of consecutive memory locations. The first is an op code. If that op code says LDA absolute the CPU knows to take the numbers in the next two memory locations as the low and high order bytes of an address. It takes the data AT THAT ADDRESS and puts it in the accumulator. Likewise, STA absolute is a 3 byte instruction using the 2 bytes following the op code to decide where to put the number in the accumulator. The sequence $8D,$55,$D8 says put the contents of the accumulator in location $D855.

The zero page addressing mode consists of two byte instructions. Since all instructions refer to zero page, ($0000-$00FF), the high order byte doesn't have to be specified, (it's $00). The two bytes $A5,$06 say take the data from location $0006 and put it in the accumulator.

The immediate mode doesn't refer to an address. $A9,$60 means take the number following the op code and put it in the accumulator, (load the accum. with $60). There is no STA immediate instruction because it makes no sense. The pound sign (#) is used to indicate the immediate mode in listings.

To get our program into memory and run it we use a Basic program called "Hexpoke", (listing 1). Hexpoke allows you to input numbers in hex and have them placed in memory. The actual numbers in memory might look like figure 1 while a listing which makes more sense is shown in listing 2. Basic grabs most of memory for itself but it turns out all of page 6 is available, so that is where we put our program. Notice in listing 2 each operation is on a seperate line along with the address, code (numbers), label and comments pertaining to the operation.

Run Hexpoke with a starting address of $0600 and you will be presented with the first 56 bytes of page 6. The four cursor keys let you move around among this data. Enter two hex keys to replace a data byte. If you make a mistake on the first, you can use the backspace. The only other keys which work are 'R' and 'ESC'. 'R' is to run the program starting at the starting address ($0600 in this case). 'ESC' is to get a new screenfull of data from another part of memory. Be sure to start your program with hex $68 (PLA) and end it with $60, (RTS). The PLA has to do with the USR function used to call the program. The RTS (Return from Subroutine) gets you back to Basic when the machine language program is finished.

Use Hexpoke to load and run the program of listing 2. Try playing around with the numbers loaded in the sound registers. You might also try a program which simply loops to itself:

```
0600  68  PLA
0601  4C 01 06 LOOP JMP LOOP
0604  60  RTS
```

You will never get to the RTS of course but you can get back to Basic with **RESET**. Here are some more things to try. Answers will appear in the next installment.

Problem #1

Use Hexpoke to find the upper left corner of the screen in your system, (use $0058 as the starting address).

The Atari has a real time clock which appears at locations $0012, $13 and $14 with $14 incrementing by one every 1/60th sec. When $14 reaches 255 it goes to zero again and $13 is also incremented. Likewise when $13 rolls over $12 is incremented.

Write a program to print out these locations in the upper left corner of the screen, (use 3 different bytes). Use zero page addressing for loads and absolute addressing for stores. Don't return to Basic with a RTS because it will clear the screen. Instead use a loop which jumps to itself and get back to Basic with RESET.

You should get different 'numbers' each time you run the program. The numbers are actually ATASCII characters and you will have to use Appendix C of the Basic Reference Manual to convert them to hex or decimal. Rather than a loop which jumps to itself, why not have the program repeat continuously by looping back to the first load instruction.

Problem #2

If you have paddle controllers: The vlaue given by paddle 0 (PADDL0) can be found in location $270. Use this value to control the pitch of voice 0. Put $AA in AUDC1 as before. If you want to continuously change the sound, you will have to loop back and pick up PADDL0 again and again.

If you have joysticks controllers: You can do the same thing with joysticks, but not as well. Joystick 0 values appear at $278. If the sounds are too high pitched for you just put 1 to 4 ASL A's (hex $0A, one byte operation), between the load and store operation. We'll talk about what this does at some later time.

LISTING 2: MIDDLE C PROGRAM

ADDRESS	CODE	LABEL	OPERATION	COMMENT
0600	68	START	PLA	; DISCARD
0601	A9 79		LDA -$79	; MIDDLE C
0603	8D 00 D2		STA AUDF1	;
0606	A9 AA		LDA -$AA	; CONTROL BYTE
0608	8D 01 D2		STA AUDC1	;
060B	60		RTS	; BACK TO BASIC

FIGURE 1:
MIDDLE C PROGRAM IN MEMORY

Memory Contents	Address (Data)
0600	68
0601	A9
0602	79
0603	8D
0604	00
0605	D2
0606	A9
0607	AA
0608	8D
0609	01
060A	D2
060B	60
060C	XX
060D	XX

TABLE 1:
SIMPLIFIED OP-CODE TABLE

	ADDRESSING MODE		
	ABSOL.	Z PAGE	IMMED.
LDA	AD	A5	A9
STA	8D	85	—
JMP	4C	—	—
	3 BYTE	2 BYTE	2 BYTE
SINGLE BYTE			
PLA	68		
RTS	60		

TINY TEXT UPDATE

by Jim Carr, Corvallis, OR

TINY TEXT is a small but very clever cassette based text editor written by Stan Ockers of Lockport, Il. The original version of TINY TEXT first appeared in the Nov. 81 issue of the A.C.E. Newsletter.

TINY TEXT was never intended to be an all purpose word processor even though it does provide several of the important features found in these larger programs. TINY TEXT was written to make it easy for ATARI users to send in "machine readable" copy of articles to the Eugene A.C.E. Newsleter. The real advantage of this program is it is small, inexpensive, and very easy to use.

The description which follows is for a slightly enhanced version of TINY TEXT. The following modifications were made by Jim Carr, Corvallis, Or:

Support for Atari 820 printer; Separate Print and Display modes; Forms control for Print mode; Top of page comm and for Print Mode; Save text on Cassette or Disk; Error trap control; Adapts to different RAM sizes. Cassette tapes recorded by the original TINY TEXT can still be used with this modified version. Finally, the modified version corrects a couple of minor formatting bugs and is about ten percent "tinier" than the original version.

USING THE PROGRAM

The OPTION key is used to select one of five options: LOAD, EDIT, PRINT, SAVE, and DISPLAY. The following paragraphs describe the use of each of these options.

LOAD OPTION: The LOAD option is used to reload text which was previously saved on cassette or disk. When the LOAD option is selected you will be asked to enter the "file name" of the text you wish to have loaded. If the text is on cassette, then simply type a "C" (no quotation marks). The computer will "beep" once to remind you to set up the recorder to play. After the recorder has been setup, press RETURN to tell the computer to begin loading the text. If the text to be loaded is on disk, type the complete file name of the text file. For example "D1:TTHELP.TXT".

EDIT OPTION:

The EDIT option lets you enter text or make changes to text previously entered.

Entering text: When the Edit mode is requested, a blank area (text entry window) appears in the center of the screen. Up to three lines of text can be typed into this window. Pressing RETURN causes text in the window to be added to previously entered text. You can use the standard screen editing functions to edit text in the window. Note: Since all trailing blanks in the window will be deleted, it is a good idea to end each entry at the end of a word and start each new entry with a space.

Functions such as tabbing and indentation are controlled by special formatting symbols. These symbols always cause the current line to be ended before the requested formatting function is executed. The following formatting symbols may be used:

CTRL E End the current line and start a new line with no indentation.
CTRL I Indent the next line.
CTRL S Space before starting the next line.
CTRL T Tab over a set number of spaces before starting the next line.
CTRL C Center the next line.
CTRL P Advance the printer forms to the top of the next page before printing the next line.

Editing Text: When you are in the edit mode, pressing the SELECT key will cause the line of text below the window to be moved up into the window. The normal screen control functions can then be used to fix up the text in the window. Use the joy stick to scroll the desired line to the position below the text window. Pressing SELECT twice without making any changes in between simply causes the text line to move up into the window and then back. To DELETE a line of text, move it below the text window and press RETURN. Pressing the joystick trigger will cause you to jump to the end of the text.

PRINT OPTION:

This option prints the formatted text on the printer. Before the printing begins you have a chance to make changes to default settings for the line length, tab stop, etc. Use the screen edit functions to make any desired change, then press **RETURN**. The items which may be changed are:

LINE - Line length (maximum number of characters per line)
INDENT - The number of spaces to be indented (left margin)
TAB STOP - The number of spaces for the tab stop
PAPER SIZE - The total number of lines which can be printed on a fully covered page. For example, 11 inch forms with 6 lines per inch have 66 lines.
FORMS FEED - The number of lines which are to be printed to seperate the bottom of one page from the top of the next. For example, if 3 blank lines are required at the top and bottom of each page, then Forms Feed will be set to 6.

SAVE OPTION:

This option lets you save text on either cassette or disk. When the SAVE option is selected you will be asked to enter the "file name" to be used in saving the text. If you are using a cassette simply type "C". The computer will beep twice to remind you to set up the recorder to record. After the recorder has been set up, press **RETURN** to begin saving text. If you wish to save the text on disk, then enter the complete file name to be used. For example "D:TTHELP.TXT".

DISPLAY OPTION:

This option displays the text on the screen. It provides the same format change options as the print option.

PROGRAMMING NOTES:

The default settings for the format control functions are defined at line 120.

If you wish to make any changes to this program you should first make a change to line 14 which automatically expands the main data storage array T$ to use all available memory. Try changing "SIZ = FRE(0)-50" to "SIZ = FRE(0)-500". When you have finished making your changes you can restore the statement to its original form.

If a system error occurs, it is trapped and printed out by the program. You are then prompted to press **RETURN** to make the program continue. When you press **RETURN**, the program will go to the OPTION selection mode. This will generally allow you to recover from errors without loss of data.

Vultures III

by Stan Ockers, Rockport, IL

The new addition in 'Vultures III' has to do with the sound. Basic tends to get slowed down if it has to continually update sound registers. Why not put the sound routines in the VBI? That's what I've done in Vultures.

I've included a listing of the machine language routine for voice #1. The same code with different registers is repeated in the VBI routine for voice #2. The data is in lines 2050-2052 and 2054-2056. Voices #3 and #4 could also be included. Eight locations at the end of page 6 hold 'string' position of current sounds being produced and the duration counts for waiting until the next change. The positions actually refer to page 6 locations. The sound 'strings' are **POKEd** into page 6.

The 'strings' consist of three byte combinations. The first is the pitch; the second a control byte made up of distortion (high nibble) and volume (low nibble). The third byte is the duration of the sound in 1/60th's of a second. I've included a Basic program 'Sounds' which helps to generate sound strings.

Use the joystick to move around the screen. Press the button and move the stick forward or back to increase or decrease the values. Press **START** to hear the sound and see a list of bytes required to make the sound. You don't have to repeat loudness and distortion bytes as long as they remain the same. The generator will repeat the same values for these two on consecutive bytes until a change is made.

Vulture movement is done using the same X & Y movement routine as in 'Balloons'. The display list has been changed and most of the Graphics 0 screen is changed to mode 4. I've only used the normal character set however because the program was already getting pretty long.

Closed, it's a compact home for your personal computer. Open the desk surface and pull out your machines. Disks are in the drawer above. Plenty of room for printer, paper storage, cassette machine. Cords and transformers are safely out of sight -- and there's even a copy holder for your work.

Atari Computer Enthusiasts
3662 Vine Maple Dr. Eugene OR 97405

FIRST CLASS MAIL

TYPESET COPY
From Your Computer

If you have a modem, word processing and communications programs, you can transmit your manuscript, article, newsletter, or other text via telephone and have it converted into space-saving, easy-to-read type! Computer typesetting offers you high quality copy at "do-it-yourself" rates: just $2.00 per 1,000 characters. And it's easy to encode your copy with our designated typesetting commands. Hundreds of type styles to choose from with 8 styles and 12 sizes "on line." Write or call Dennis Hunt for a brochure and simple instructions.

EDITING & DESIGN SERVICES
30 East 13th Avenue Eugene, Oregon 97401
Phone (503) 683-2657

Atari Computer Enthusiasts

A.C.E. is an independent computer club and user's group with no connection to the Atari Company, a division of Warner Communication Company. We are a group interested in educating our members in the use of the Atari Computer and in giving the latest News, Reviews and Rumors.

All our articles, reviews and programs come from you, our members.

Our membership is world-wide in scope; membership fees include the A.C.E. Newsletter. Dues are $10 a year for U.S., and $20 a year Overseas Airmail and include about 10 issues a year of the ACE Newsletter.

President—Kirt Stockwell, 1810 Harris #139, Eugene, Or 97403 503-686-2470
Secretary—Charles Andrews, POB 1613, Eugene, Or 974401613 503-747-9892
Librarian—Chuck and Jody Ross, 2222 Ironwood, Eugene, OR 97401 503-343-5545
Editors—Mike Dunn, 3662 Vine Maple Dr., Eugene, Or 97405 503-344-6193
Jim Bumpas, 4405 Dillard Rd., Eugene, Or 97405 503-484-9925

Send a business-size SASE to the Ross' for the new, updated ACE Library List!! Best of ACE-1981, still available, Disk or Tape, $8!